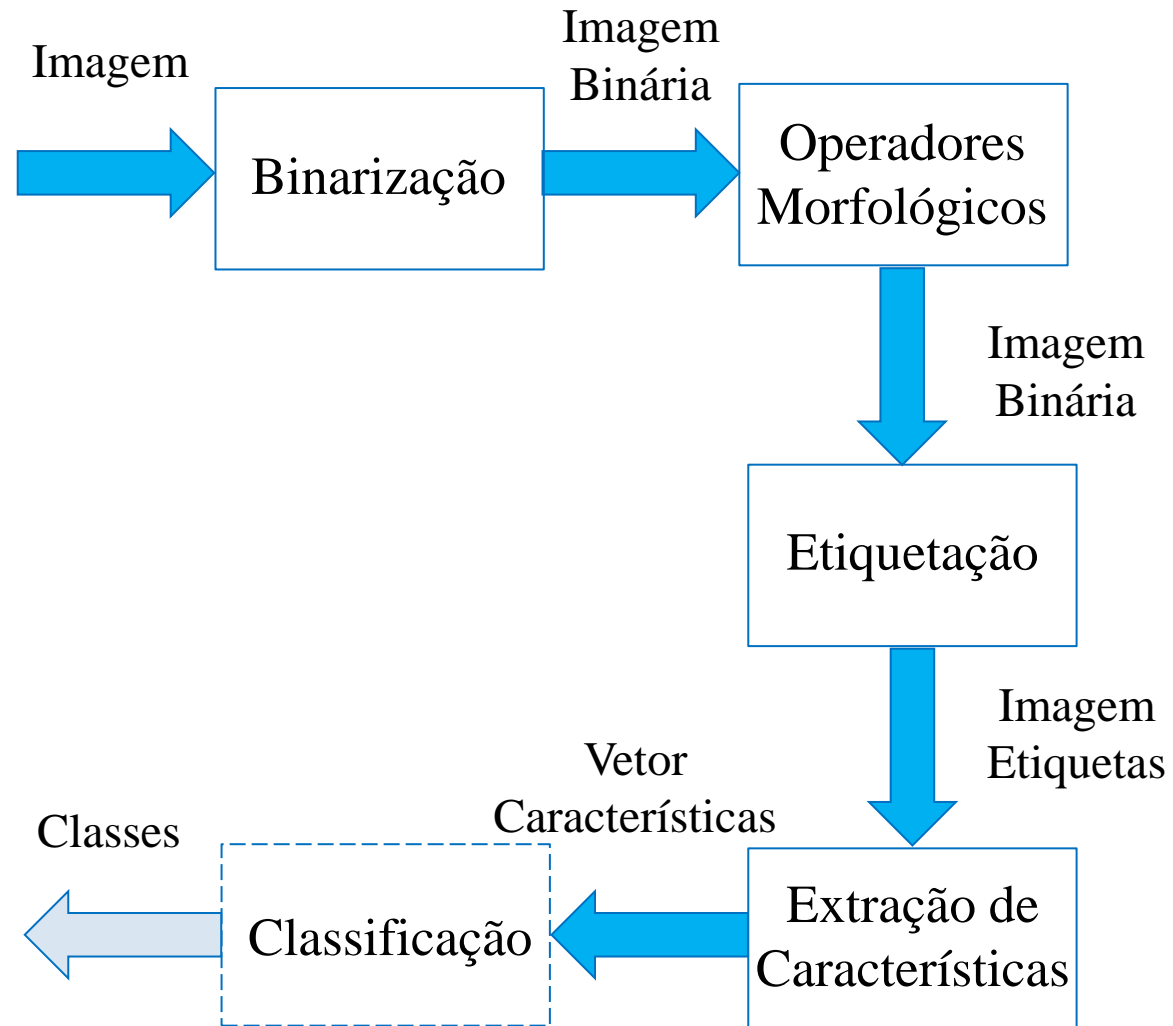


3º CAPÍTULO

Análise de Imagens Binárias



Prof. Arnaldo Abrantes



- Vizinhanças mais comuns

	N	
W	*	E
	S	

Vizinhança N_4

NW	N	NE
W	*	E
SW	S	SE

Vizinhança N_8

- Utilização de máscaras

– Exemplo:

1	1	1
1	1	1
1	1	1

1	2	1
2	4	2
1	2	1

1
1
1
1
1

origem

40	40	80	80	80
40	40	80	80	80
40	40	80	80	80
40	40	80	80	80
40	40	80	80	80

entrada →



40	50	70	80	80
40	50	70	80	80
40	50	70	80	80
40	50	70	80	80
40	50	70	80	80

saída →



- Algoritmo
 - **Hipótese:** Objecto é um conjunto conexo de pixels (conectividade 4) e sem buracos no seu interior

0	0	0	0	1	0	0	1
0	1	1	0	0	0	0	0

Cantos exteriores

1	1	1	1	1	0	0	1
1	0	0	1	1	1	1	1

Cantos interiores

Compute the number of foreground objects of binary image B.

Objects are 4-connected and simply connected.

E is the number of external corners.

I is the number of internal corners.

```
procedure count_objects(B);
{
  E := 0;
  I := 0;
  for L := 0 to MaxRow - 1
    for P := 0 to MaxCol - 1
      {
        if external_match(L, P) then E := E + 1;
        if internal_match(L, P) then I := I + 1;
      };
  return((E - I) / 4);
}
```

Quantos objectos?

							e	e	
					e		i		
e			e						
	i	i			e			e	
e	e				e	e			
e		i			e	e			
e			e				e	e	
						e	i		
						e		e	

e - 21

i - 5

$$\# = \frac{21-5}{4} = \frac{16}{4} = 4$$

E agora?

							e	e	
					e		i		
e			e						
	i	i			e			e	
e	e				e	e			
e		i			e	e			
e				e			e	e	
			e	e		e	i		
						e		e	

e - 23

i - 4

$$\# = \frac{23 - 4}{4} = \frac{19}{4} = ?$$

Extracção de componentes conexos - algoritmo recursivo

Compute the connected components of a binary image.

B is the original binary image.

LB will be the labeled connected component image.

```
procedure recursive_connected_components(B, LB);  
{  
  LB := negate(B);  
  label := 0;  
  find_components(LB, label);  
  print(LB);  
}
```

```
procedure find_components(LB, label);  
{  
  for L := 0 to MaxRow  
    for P := 0 to MaxCol  
      if LB[L,P] == -1 then  
        {  
          label := label + 1;  
          search(LB, label, L, P);  
        }  
}
```

```
procedure search(LB, label, L, P);  
{  
  LB[L,P] := label;  
  Nset := neighbors(L, P);  
  for each (L',P') in Nset  
    {  
      if LB[L',P'] == -1  
        then search(LB, label, L', P');  
    }  
}
```

1	1	0	1	1	1	0	1
1	1	0	1	0	1	0	1
1	1	1	1	0	0	0	1
0	0	0	0	0	0	0	1
1	1	1	1	0	1	0	1
0	0	0	1	0	1	0	1
1	1	0	1	0	0	0	1
1	1	0	1	0	1	1	1

↑
entrada

saída
↓

1	1	0	1	1	1	0	2
1	1	0	1	0	1	0	2
1	1	1	1	0	0	0	2
0	0	0	0	0	0	0	2
3	3	3	3	0	4	0	2
0	0	0	3	0	4	0	2
5	5	0	3	0	0	0	2
5	5	0	3	0	2	2	2



Algoritmo recursivo - Exemplo

Step 1.

-1	-1	0	-1	-1	-1
-1	-1	0	-1	0	0
-1	-1	-1	-1	0	0

Step 2.

1	-1	0	-1	-1	-1
-1	-1	0	-1	0	0
-1	-1	-1	-1	0	0

Step 3.

1	1	0	-1	-1	-1
-1	-1	0	-1	0	0
-1	-1	-1	-1	0	0

Step 4.

1	1	0	-1	-1	-1
1	-1	0	-1	0	0
-1	-1	-1	-1	0	0

Step 5.

1	1	0	-1	-1	-1
1	1	0	-1	0	0
-1	-1	-1	-1	0	0

Vizinhança 4

	1	
2	*	3
	4	

Vizinhança 8

1	2	3
4	*	5
6	7	8

Estrutura União-Procura

Construct the union of two sets.

X is the label of the first set.

Y is the label of the second set.

PARENT is the array containing the union-find data structure.

```
procedure union(X, Y, PARENT);
{
  j := X;
  k := Y;
  while PARENT[j] <> 0
    j := PARENT[j];
  while PARENT[k] <> 0
    k := PARENT[k];
  if j <> k then PARENT[k] := j;
}
```

Find the parent label of a set.

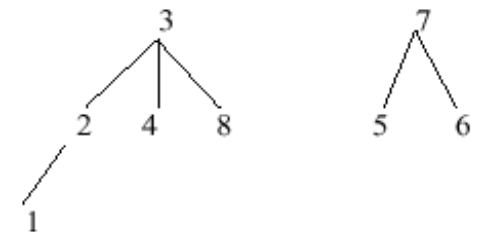
X is a label of the set.

PARENT is the array containing the union-find data structure.

```
procedure find(X, PARENT);
{
  j := X;
  while PARENT[j] <> 0
    j := PARENT[j];
  return(j);
}
```

PARENT

1	2	3	4	5	6	7	8
2	3	0	3	7	7	0	3



ECC - algoritmo clássico com união-procura

Compute the connected components of a binary image.

B is the original binary image.

LB will be the labeled connected component image.

```

procedure classicalwithunionfind(B, LB);
{
  "Initialize structures."
  initialize();
  "Pass 1 assigns initial labels to each row L of the image."
  for L := 0 to MaxRow
  {
    "Initialize all labels on line L to zero"
    for P := 0 to MaxCol
      LB[L,P] := 0;
    "Process line L."
    for P := 0 to MaxCol
      if B[L,P] == 1 then
      {
        A := prior_neighbors(L,P);
        if isempty(A)
          then { M := label; label := label + 1; };
        else M := min(labels(A));
        LB[L,P] := M;
        for X in labels(A) and X <> M
          union(M, X, PARENT);
      }
  }
  "Pass 2 replaces Pass 1 labels with equivalence class labels."
  for L := 0 to MaxRow
    for P := 0 to MaxCol
      if B[L,P] == 1
        then LB[L,P] := find(LB[L,P], PARENT);
  };

```

1	1	0	1	1	1	0	1
1	1	0	1	0	1	0	1
1	1	1	1	0	0	0	1
0	0	0	0	0	0	0	1
1	1	1	1	0	1	0	1
0	0	0	1	0	1	0	1
1	1	0	1	0	0	0	1
1	1	0	1	0	1	1	1

← entrada

1º passo →

1	1	0	2	2	2	0	3
1	1	0	2	0	2	0	3
1	1	1	1	0	0	0	3
0	0	0	0	0	0	0	3
4	4	4	4	0	5	0	3
0	0	0	4	0	5	0	3
6	6	0	4	0	0	0	3
6	6	0	4	0	7	7	3

PARENT

1	2	3	4	5	6	7
0	1	0	0	0	0	3

← classes equiv.

2º passo →

1	1	0	1	1	1	0	3
1	1	0	1	0	1	0	3
1	1	1	1	0	0	0	3
0	0	0	0	0	0	0	3
4	4	4	4	0	5	0	3
0	0	0	4	0	5	0	3
6	6	0	4	0	0	0	3
6	6	0	4	0	3	3	3

Exercício

- Aplique o algoritmo de ECC, considerando conectividade 4

	1			2					
	1	1	1	1	1				
			1		1				
	3	3	1						
		3						4	
		3	3				5	4	
		3							

1	2	3	4	5
0	0	0	0	0

- Qual deveria ser a forma da máscara se considerasse conectividade 8?

- Elementos estruturantes

<table><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr></table>	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	<table><tr><td></td><td>1</td><td>1</td><td>1</td><td></td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr><tr><td></td><td>1</td><td>1</td><td>1</td><td></td></tr></table>		1	1	1		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1		1	1	1		<table><tr><td></td><td>1</td><td>1</td><td>1</td><td></td></tr><tr><td>1</td><td></td><td></td><td></td><td>1</td></tr><tr><td>1</td><td></td><td></td><td></td><td>1</td></tr><tr><td>1</td><td></td><td></td><td></td><td>1</td></tr><tr><td></td><td>1</td><td>1</td><td>1</td><td></td></tr></table>		1	1	1		1				1	1				1	1				1		1	1	1		<table><tr><td>1</td><td>1</td><td></td><td></td></tr><tr><td>1</td><td>1</td><td></td><td></td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td><td>1</td></tr></table>	1	1			1	1			1	1	1	1	1	1	1	1	<table><tr><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td><td>1</td></tr><tr><td>1</td><td></td><td>1</td><td>1</td><td></td><td>1</td></tr><tr><td>1</td><td></td><td>1</td><td>1</td><td></td><td>1</td></tr><tr><td>1</td><td></td><td>1</td><td>1</td><td></td><td>1</td></tr></table>	1	1	1	1	1	1	1		1	1		1	1		1	1		1	1		1	1		1	<table><tr><td>1</td></tr><tr><td>1</td></tr><tr><td>1</td></tr><tr><td>1</td></tr></table>	1	1	1	1
1	1	1	1	1																																																																																																														
1	1	1	1	1																																																																																																														
1	1	1	1	1																																																																																																														
	1	1	1																																																																																																															
1	1	1	1	1																																																																																																														
1	1	1	1	1																																																																																																														
1	1	1	1	1																																																																																																														
	1	1	1																																																																																																															
	1	1	1																																																																																																															
1				1																																																																																																														
1				1																																																																																																														
1				1																																																																																																														
	1	1	1																																																																																																															
1	1																																																																																																																	
1	1																																																																																																																	
1	1	1	1																																																																																																															
1	1	1	1																																																																																																															
1	1	1	1	1	1																																																																																																													
1		1	1		1																																																																																																													
1		1	1		1																																																																																																													
1		1	1		1																																																																																																													
1																																																																																																																		
1																																																																																																																		
1																																																																																																																		
1																																																																																																																		
ones(3,5)	disco(5)	anel(5)																																																																																																																

– necessário definir uma origem

- Definição:** A **dilatação** duma imagem binária B pelo elemento estruturante S define-se da seguinte forma

$$B \oplus S = \bigcup_{b \in B} S_b \quad S_b = \{s + b | s \in S\}$$

- Definição:** A **erosão** duma imagem binária B pelo elemento estruturante S define-se da seguinte forma

$$B \ominus S = \{b | b + s \in B \forall s \in S\}$$

Operadores morfológicos - Exemplos

1	1	1	1	1	1	1	
			1	1	1	1	
			1	1	1	1	
		1	1	1	1	1	
			1	1	1	1	
		1	1				

a) Binary image B

1	1	1
1	1	1
1	1	1

b) Structuring Element S

1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
	1	1	1	1	1	1	1
	1	1	1	1	1	1	1
	1	1	1	1	1	1	1
	1	1	1	1	1	1	1
	1	1	1	1			

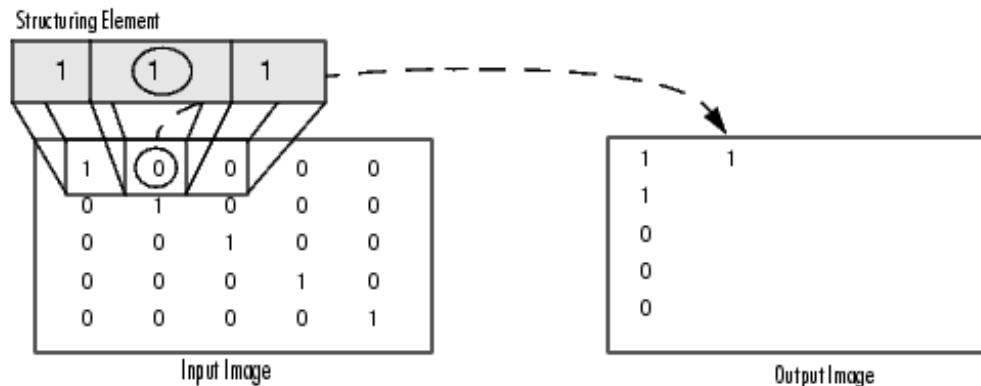
c) Dilation $B \oplus S$

				1	1		
				1	1		
				1	1		

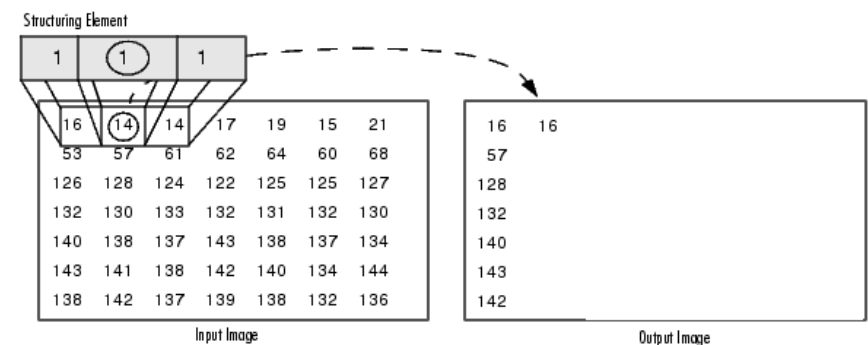
d) Erosion $B \ominus S$

Dilatação e erosão – Generalização para imagens monocromáticas

Operação	Regra
Dilatação	O valor do pixel de saída é o valor máximo de todos os pixels na vizinhança do pixel de entrada. É atribuído o valor mínimo (0) aos pixels exteriores
Erosão	O valor do pixel de saída é o valor mínimo de todos os pixels na vizinhança do pixel de entrada. É atribuído o valor máximo (1 ou 255) aos pixels exteriores



Dilatação de imagem binária



Dilatação de imagem monocromática

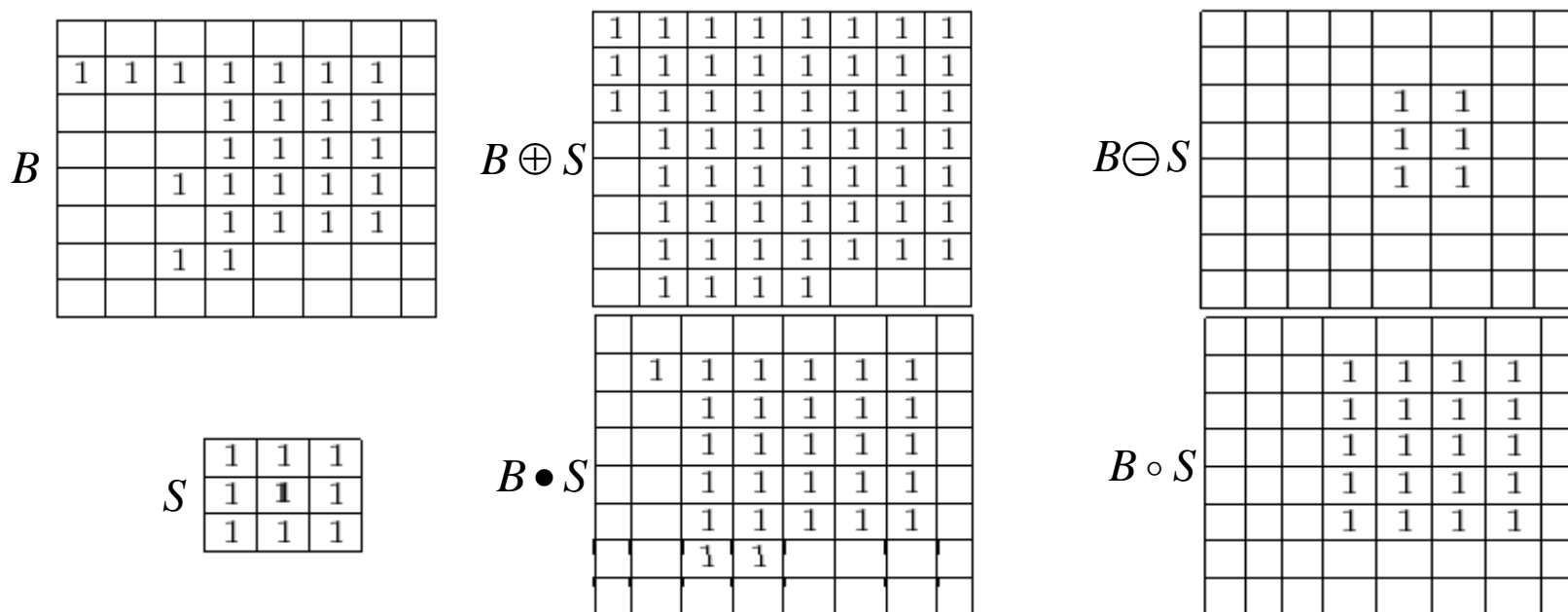
Operadores morfológicos

- **Definição:** O **fecho** duma imagem binária B pelo elemento estruturante S define-se da seguinte forma

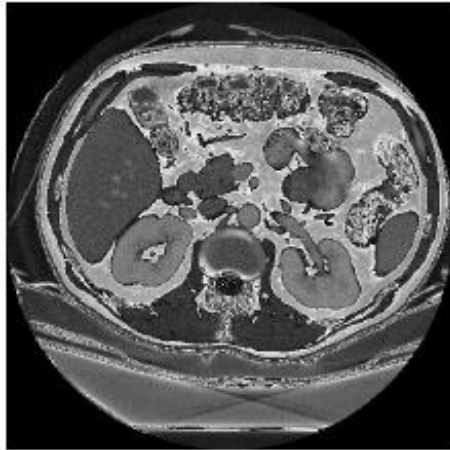
$$B \bullet S = (B \oplus S) \ominus S$$

- **Definição:** A **abertura** duma imagem binária B pelo elemento estruturante S define-se da seguinte forma

$$B \circ S = (B \ominus S) \oplus S$$



- aplicações médicas (resolução 512x512)
 - **abertura** com disco(13) seguido de **fecho** com disco(2)



original



binarizada

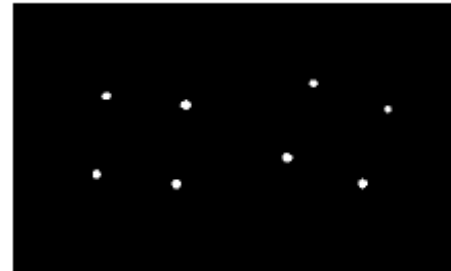
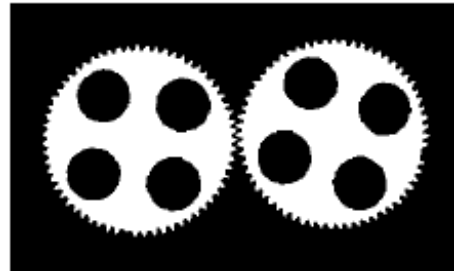


processada

- extracção de primitivas geométricas
 - subtrai da imagem original a obtida desta através do operador **abertura** usando pequeno disco como elemento estruturante



entrada →

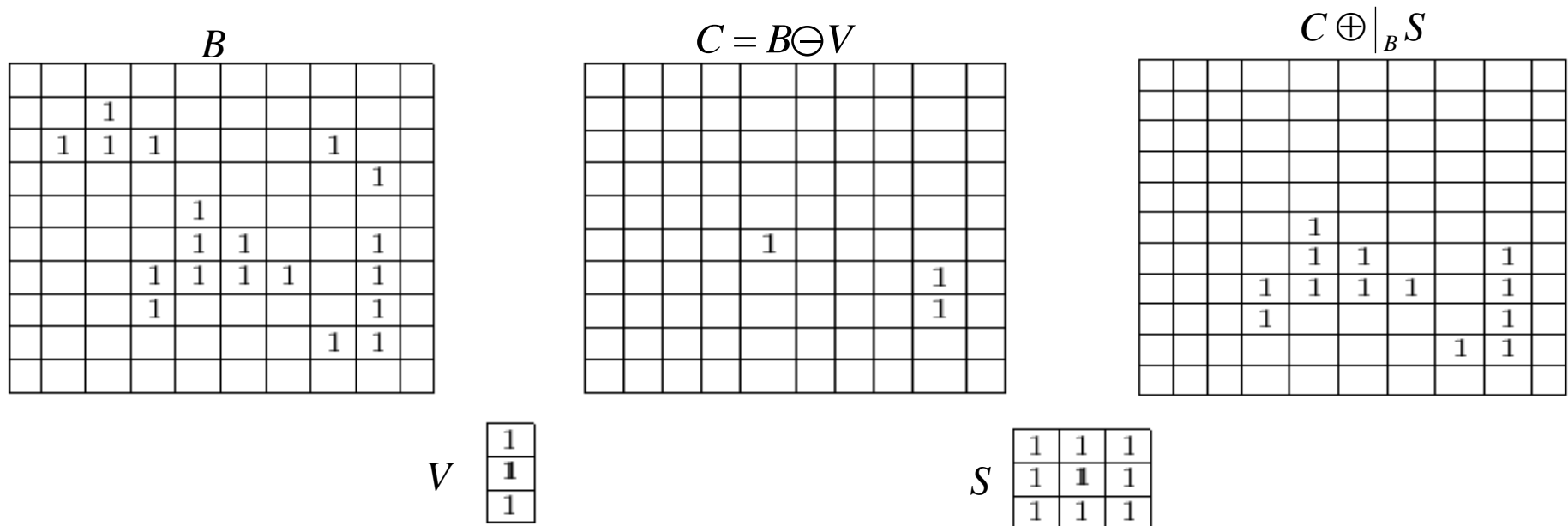


← saída

Dilatação condicional

- **Definição:** Dadas as imagens binárias original B , e processada C , e o elemento estruturante S , e seja $C_0 = C$ e $C_n = (C_{n-1} \oplus S) \cap B$. A **dilatação condicional** de C por S com respeito a B define-se como $C \oplus|_B S = C_m$

onde m é o menor inteiro que satisfaz a condição $C_m = C_{m-1}$



- Área

$$A = \sum_{(r,c) \in R} 1$$

- Centróide

$$\bar{r} = \frac{1}{A} \sum_{(r,c) \in R} r \quad \bar{c} = \frac{1}{A} \sum_{(r,c) \in R} c$$

- Pixels de perímetro

$$P_4 = \{(r,c) \in R \mid N_8(r,c) - R \neq \emptyset\}$$

$$P_8 = \{(r,c) \in R \mid N_4(r,c) - R \neq \emptyset\}$$

- comprimento do perímetro

$$|P| = \left| \{k \mid (r_{k+1}, c_{k+1}) \in N_4(r_k, c_k)\} \right| \\ + \sqrt{2} \left| \{k \mid (r_{k+1}, c_{k+1}) \in N_8(r_k, c_k) - N_4(r_k, c_k)\} \right|$$

- Circularidade (1)

$$C_1 = \frac{|P|^2}{A}$$

- Circularidade (2)

$$C_2 = \frac{\mu_R}{\sigma_R}$$

- distância radial média

$$\mu_R = \frac{1}{K} \sum_{k=0}^{K-1} \|(r_k, c_k) - (\bar{r}, \bar{c})\|$$

- desvio padrão da distância radial

$$\sigma_R = \left(\frac{1}{K} \sum_{k=0}^{K-1} (\|(r_k, c_k) - (\bar{r}, \bar{c})\| - \mu_R)^2 \right)^{1/2}$$

```

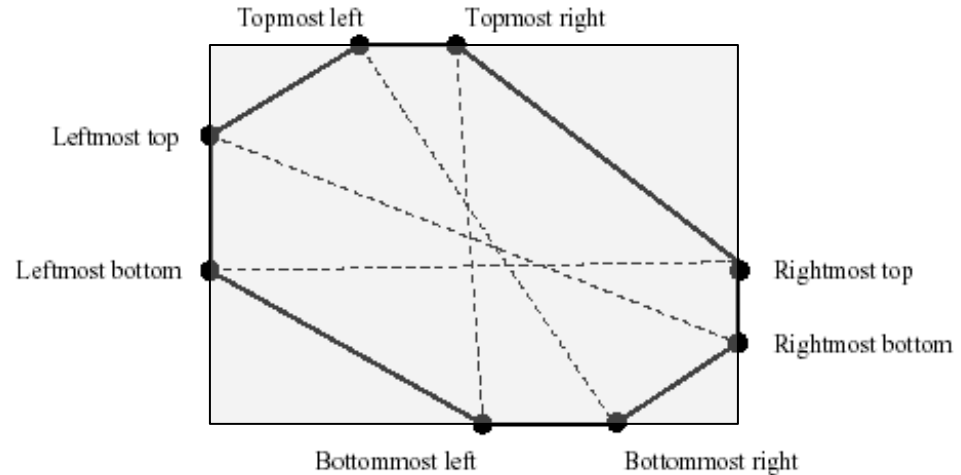
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 1 1 1 1 0
2 2 2 2 0 0 0 0 0 1 1 1 1 1 0
2 2 2 2 0 0 0 0 1 1 1 1 1 1 1
2 2 2 2 0 0 0 0 1 1 1 1 1 1 1
2 2 2 2 0 0 0 0 1 1 1 1 1 1 1
2 2 2 2 0 0 0 0 0 1 1 1 1 1 0
2 2 2 2 0 0 0 0 0 1 1 1 1 0 0
2 2 2 2 0 0 0 0 0 0 0 0 0 0 0
2 2 2 2 0 0 0 0 0 0 0 0 0 0 0
2 2 2 2 0 0 3 3 3 0 0 0 0 0 0
2 2 2 2 0 0 3 3 3 0 0 0 0 0 0
2 2 2 2 0 0 3 3 3 0 0 0 0 0 0
2 2 2 2 0 0 0 0 0 0 0 0 0 0 0

```

region num.	region area	row of center	col of center	perim. length	circu- larity ₁	circu- larity ₂	radius mean	radius var.
1	44	6	11.5	21.2	10.2	15.4	3.33	.05
2	48	9	1.5	28	16.3	2.5	3.80	2.28
3	9	13	7	8	7.1	5.8	1.2	0.04

Propriedades – fronteiras e comprimentos

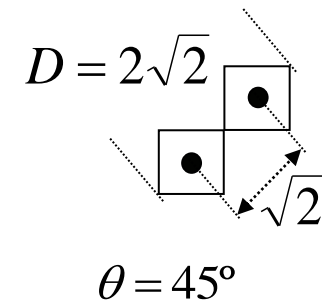
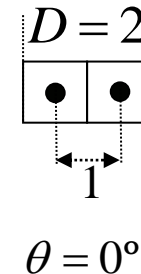
- Rectângulo (e octógono) de fronteira



- Comprimento de um segmento (eixo)

$$D = \sqrt{(r_2 - r_1)^2 + (c_2 - c_1)^2} + Q(\theta)$$

$$Q(\theta) = \begin{cases} \frac{1}{|\cos(\theta)|} & : |\theta| < 45^\circ \\ \frac{1}{|\sin(\theta)|} & : |\theta| > 45^\circ \end{cases}$$



Propriedades – Momentos de 2ª ordem

- Momentos de 2ª ordem centrados

$$\mu_{rr} = \frac{1}{A} \sum_{(r,c) \in R} (r - \bar{r})^2 \quad \mu_{cc} = \frac{1}{A} \sum_{(r,c) \in R} (c - \bar{c})^2 \quad \mu_{rc} = \frac{1}{A} \sum_{(r,c) \in R} (r - \bar{r})(c - \bar{c})$$

- Relação entre momentos e regiões elípticas

$$R = \{(r, c) \mid dr^2 + 2erc + fc^2 \leq 1\} \quad \begin{pmatrix} d & e \\ e & f \end{pmatrix}_c = \frac{1}{4(\mu_{rr}\mu_{cc} - \mu_{rc}^2)} \begin{pmatrix} \mu_{cc} & -\mu_{rc} \\ -\mu_{rc} & \mu_{rr} \end{pmatrix}$$

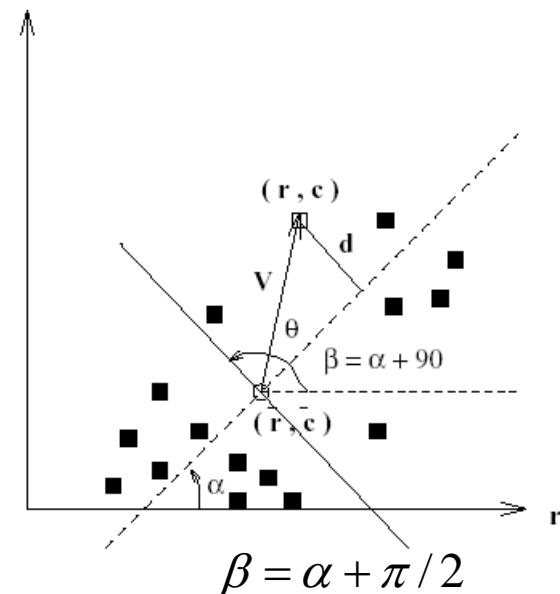
- Eixos de menor e maior inércia

– Formulação

$$\begin{aligned} \mu_{\bar{r}, \bar{c}, \alpha} &= \frac{1}{A} \sum_{(r,c) \in R} d^2 \\ &= \frac{1}{A} \sum_{(r,c) \in R} (\bar{V} \circ (\cos \beta, \sin \beta))^2 \end{aligned}$$

– Solução

$$\tan(2\hat{\alpha}) = \frac{2\mu_{rc}}{\mu_{rr} - \mu_{cc}}$$

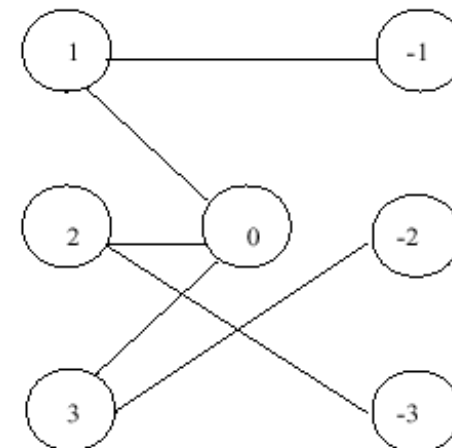


Grafos de adjacências de regiões

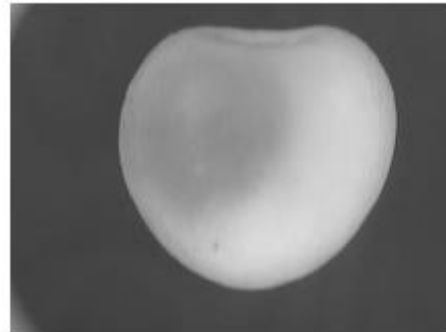
- **Problema:** regiões possuem buracos (fundo) no seu interior
- **Solução:** algoritmo com 3 passos
 - aplicação do algoritmo de extracção de componentes conexos duas vezes: (1) aos pixels activos e (2) aos pixels do fundo

0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	0	2	2	0
0	1	-1	-1	-1	1	0	2	2	0
0	1	1	1	1	1	0	2	2	0
0	0	0	0	0	0	0	2	2	0
0	3	3	3	0	2	2	2	2	0
0	3	-2	3	0	2	-3	-3	2	0
0	3	-2	3	0	2	-3	-3	2	0
0	3	3	3	0	2	2	2	2	0
0	0	0	0	0	0	0	0	0	0

- (3) construção de grafo de relações



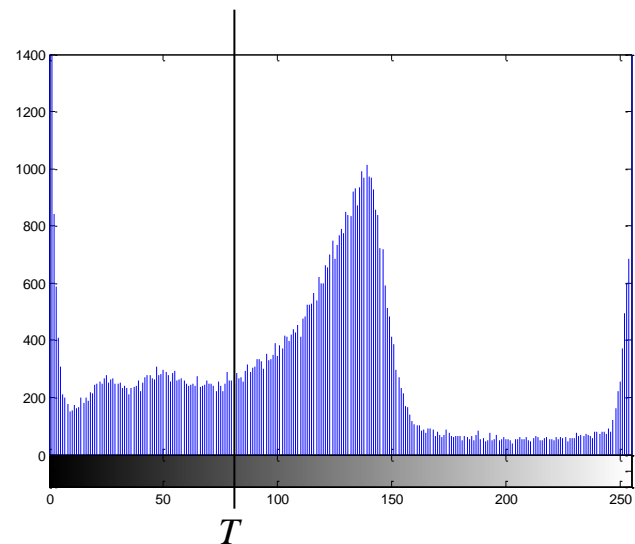
- **Definição:** o histograma h da imagem monocromática I é definido por
$$h(m) = |\{(r,c) | I(r,c) = m\}|$$



Compute the histogram H of gray-tone image I .

```
procedure histogram(I,H);  
{  
  "Initialize the bins of the histogram to zero."  
  for i := 0 to MaxVal  
    H[i] := 0;  
  "Compute values by accumulation."  
  for L := 0 to MaxRow  
    for P := 0 to MaxCol  
      {  
        grayval := I[r,c];  
        H[grayval] := H[grayval] + 1;  
      };  
}
```

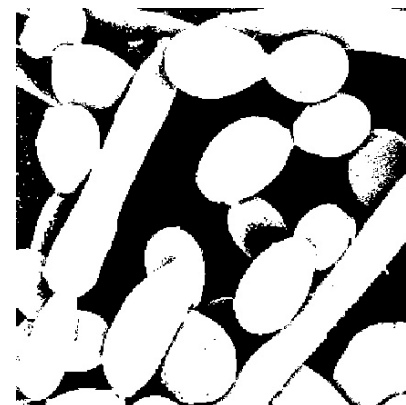
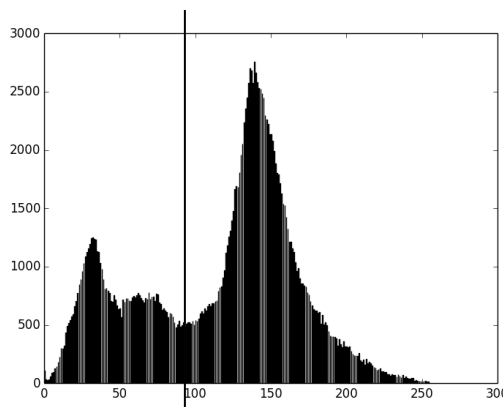
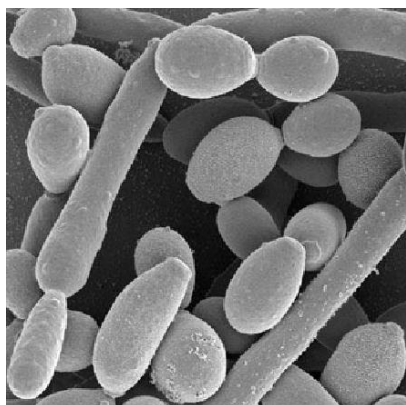
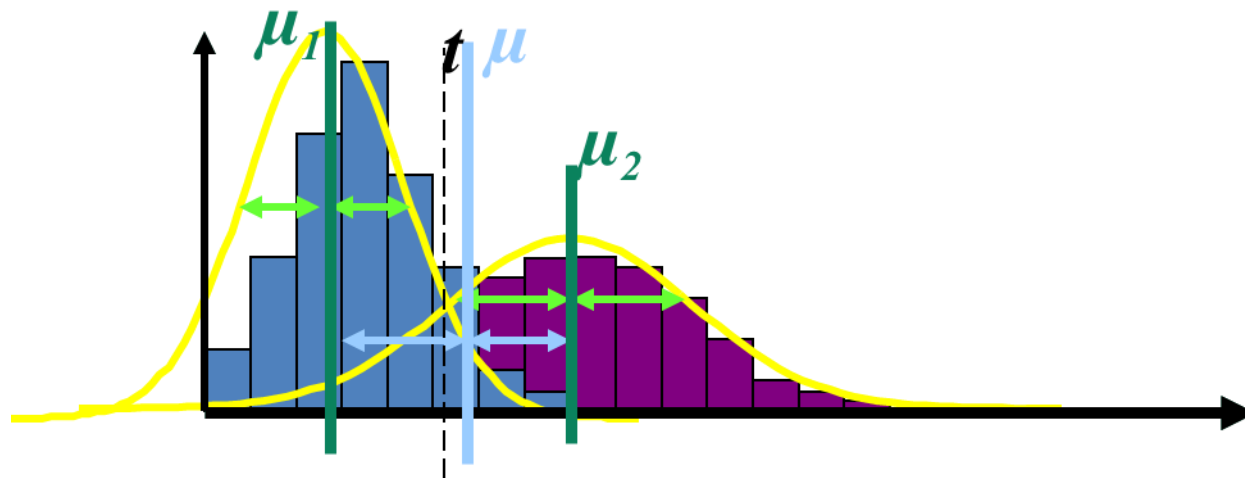

Binarização por Limiar Global



$$g(x, y) = \begin{cases} 1 & \text{se } f(x, y) > T \\ 0 & \text{se } f(x, y) \leq T \end{cases}$$



- Método de Otsu



$T=99$

- Método de Otsu

- **Ideia:** minimização da variância intra-classes $\sigma_W^2 = q_1(t)\sigma_1^2(t) + q_2(t)\sigma_2^2(t)$

$$q_1(t) = \sum_{i=0}^t P(i)$$

$$q_2(t) = \sum_{i=t+1}^{MaxVal} P(i)$$

$$\sigma_1^2(t) = \sum_{i=0}^t (i - \mu_1(t))^2 P(i) / q_1(t)$$

$$\sigma_2^2(t) = \sum_{i=t+1}^{MaxVal} (i - \mu_2(t))^2 P(i) / q_2(t)$$

$$\mu_1(t) = \sum_{i=0}^t iP(i) / q_1(t)$$

$$\mu_2(t) = \sum_{i=t+1}^{MaxVal} iP(i) / q_2(t)$$

$$P(i) = h(i) / |R \times C|$$

- Nota: equivalente à maximização da variância inter-classes

$$\sigma_B^2 = q_1(t)(1 - q_1(t))(\mu_1(t) - \mu_2(t))^2$$

$$\sigma^2 = \sigma_W^2 + \sigma_B^2$$

- Algoritmo de Otsu para determinação do limiar t

- Iniciar

$$P(i) = h(i) / |R \times C| \quad q_1(0) = P(0)$$

$$\mu = \sum_{i=0}^{MaxVal} iP(i) \quad \mu_1(0) = 0$$

- For $t := 0$ to $MaxVal-1$

$$q_1(t+1) = q_1(t) + P(t+1)$$

$$\mu_1(t+1) = \frac{q_1(t)\mu_1(t) + (t+1)P(t+1)}{q_1(t+1)}$$

$$\mu_2(t+1) = \frac{\mu - q_1(t+1)\mu_1(t+1)}{1 - q_1(t+1)}$$

$$\sigma_B^2(t) = q_1(t)(1 - q_1(t))(\mu_1(t) - \mu_2(t))^2$$

- Determinação do limiar

$$\hat{t} = \arg \max_t \sigma_B^2(t)$$

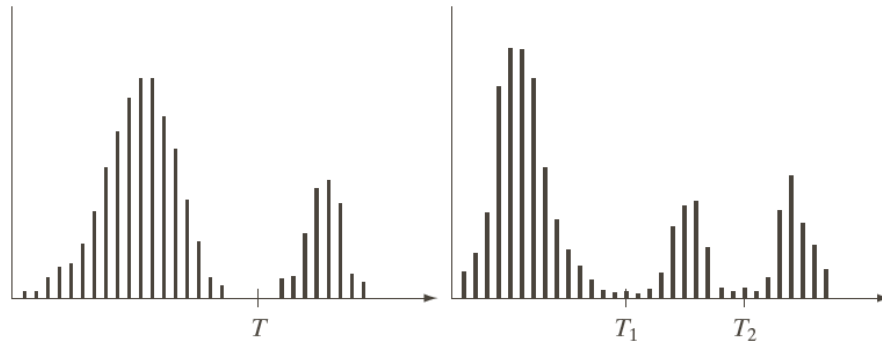


Original ($MaxVal=255$)



$t = 93$

Multiple Thresholding



$$g(x, y) = \begin{cases} a & \text{se} & f(x, y) \geq T_2 \\ b & \text{se} & T_1 \leq f(x, y) < T_2 \\ c & \text{se} & f(x, y) < T_1 \end{cases}$$

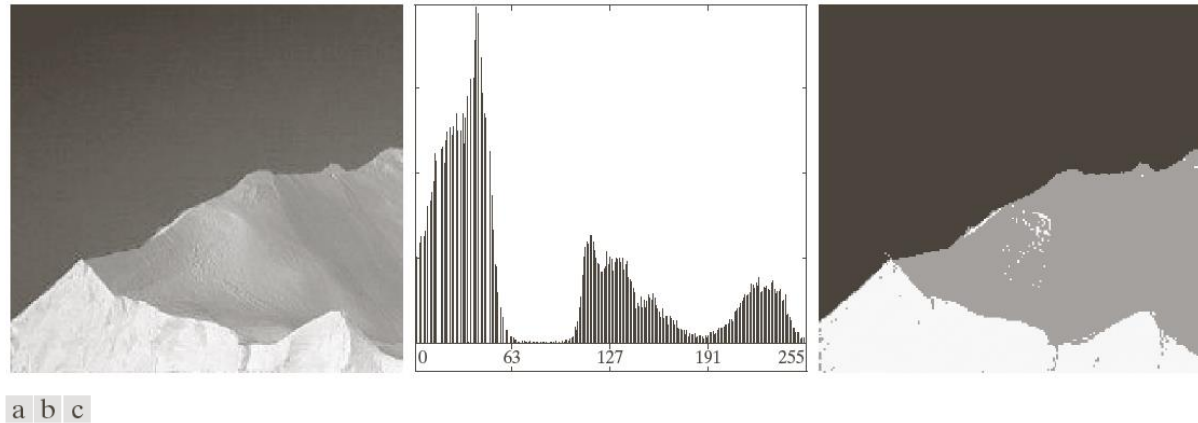


FIGURE 10.45 (a) Image of iceberg. (b) Histogram. (c) Image segmented into three regions using dual Otsu thresholds. (Original image courtesy of NOAA.)

- Técnicas de melhoramento
 - Filtragem
 - Utilização dos contornos
- Outras técnicas
 - Limiar variável (partição da imagem)
 - Limiar local baseado em vizinhos